	Type	L #	Hits	Search Text	DBs	Time Stamp
1	BRS	L1	6	legend and component and (system or building or management or support) and ((texture or color or shading) near coding) and @pd>20011201	USPAT	2002/06/06 14:29 Considered all
2	BRS	L2	5	legend and component and (system or building or management or support) and ((texture or color or shading) near coding)	US-PGPUB	2002/06/06 14:59 considered all
3	BRS	L3	106	component and (system or building or management or support) and (legend and ((texture or color or shading) near coding))	USPAT	2002/06/06 15:04 Considered new patents
4	BRS	L4	14502	(component or components or parts) and (system or building or management or support) and (legend or ((texture or color or shading) near coding))	USPAT	2002/06/06 15:07
5	BRS	L5	146	(component or components or parts) and (system or building or management or support) and (legend and ((texture or color or shading) near coding))	USPAT	2002/06/06 15:08 considered [illegible]

```
### Status: Path 1 of [Dialog Information Services via Modem]

### Status: Initializing TCP/IP using (UseTelnetProto 1 ServiceID pto-dialog)
Trying 3106900061...Open

DIALOG INFORMATION SERVICES
PLEASE LOGON:
 ******** HHHHHHHH SSSSSSSS?
### Status: Signing onto Dialog
 ********
ENTER PASSWORD:
 ******** HHHHHHHH SSSSSSSS? ********
Welcome to DIALOG
### Status: Connected


Dialog level 01.10.01D

Last logoff:  14nov01 13:40:57
Logon file405 03dec01 14:39:44
           *** ANNOUNCEMENT ***
                   ***
--Important Notice to Freelance Authors--
See HELP FREELANCE for more information
                   ***
NEW FILES RELEASED
***Disclosure Database (File 101)
***Harris Business Profiler (File 537)
***Mergent Company Profiles (File 555)
***Mergent Company Snapshots (File 556)
***Mergent Company News Reports (File 557)
***Financial Times Fulltext (File 476)

***TRADEMARKSCAN-Japan (File 669)

UPDATING RESUMED
***Delphes European Business (File 481)
***Books In Print (File 470)
                           ***
RELOADED
***CLAIMS/US PATENTS (Files 340, 341, 942)
***Kompass Middle East/Africa/Mediterranean (File 585)
***Kompass Asia/Pacific (File 592)
***Kompass Central/Eastern Europe (File 593)
***Kompass Canada (File 594)

***New document supplier***
IMED has been changed to INFOTRIE (see HELP OINFOTRI)


>>>Get immediate news with Dialog's First Release
   news service.  First Release updates major newswire
   databases within 15 minutes of transmission over the
   wire.  First Release provides full Dialog searchability
   and full-text features.  To search First Release files in
   OneSearch simply BEGIN FIRST for coverage from Dialog's
   broad spectrum of news wires.

     >>> Enter BEGIN HOMEBASE for Dialog Announcements <<<
     >>>    of new databases, price changes, etc.      <<<
                   ****
COREFULL is set ON as an alias for 15,9,623,810,275,624,636,621,813,16,160,148,20.
COREABS is set ON as an alias for 77,35,593,65,2,233,99,473,474,475.
COREALL is set ON as an alias for COREFULL,COREABS.
SOFTFULL is set ON as an alias for 278,634,256.
EUROFULL is set ON as an alias for 348,349.
```

```
JAPOABS is set ON as an alias for 347.
HEALTHFULL is set ON as an alias for 442,149,43,444.
HEALTHABS is set ON as an alias for 5,73,151,155,34,434.
DRUGFULL is set ON as an alias for 455,129,130.
DRUGABS is set ON as an alias for 74,42.
INSURANCEFULL is set ON as an alias for 625,637.
INSURANCEABS is set ON as an alias for 169.
TRANSPORTFULL is set ON as an alias for 80,637.
TRANSPORTABS is set ON as an alias for 108,6,63.
ADVERTISINGFULL is set ON as an alias for 635,570,PAPERSMJ,PAPERSEU.
INVENTORYABS is set ON as an alias for 8,14,94,6,34,434,7.
BANKINGFULL is set ON as an alias for 625,268,626,267.
BANKINGABS is set ON as an alias for 139.
HEALTHALL is set ON as an alias for COREFULL,COREABS,HEALTHFULL,HEALTHABS.
INSURANCEALL is set ON as an alias for COREFULL,COREABS,INSURANCEFULL,INSURANCEABS.
RESERVATIONALL is set ON as an alias for COREFULL, COREABS.
OPERATIONSALL is set ON as an alias for COREFULL,COREABS,INVENTORYABS.
TRANSPORTALL is set ON as an alias for COREFULL,COREABS,TRANSPORTFULL,TRANSPORTABS.
ADVERTISINGALL is set ON as an alias for COREFULL,COREABS,ADVERTISINGFULL.
SHOPPINGALL is set ON as an alias for COREFULL,COREABS,ADVERTISINGALL,47.
INVENTORYALL is set ON as an alias for COREFULL,COREABS,INVENTORYFULL.
BANKINGALL is set ON as an alias for COREFULL,COREABS,BANKINGFULL,BANKINGABS.
PORTFOLIOALL is set ON as an alias for COREFULL,COREABS,BANKINGALL.
TRADINGALL is set ON as an alias for COREFULL,COREABS,BANKINGALL.
CREDITALL is set ON as an alias for COREFULL,COREABS,BANKINGALL.
FUNDSALL is set ON as an alias for COREFULL,COREABS,BANKINGALL,608.
****                                   *****
SYSTEM:HOME
Cost is in DialUnits
Menu System II: D2 version 1.7.8 term=ASCII
                     *** DIALOG HOMEBASE(SM) Main Menu ***

 Information:
  1.  Announcements (new files, reloads, etc.)
  2.  Database, Rates, & Command Descriptions
  3.  Help in Choosing Databases for Your Topic
  4.  Customer Services (telephone assistance, training, seminars, etc.)
  5.  Product Descriptions

 Connections:
  6.  DIALOG(R) Document Delivery
  7.  Data Star(R)

    (c) 2000  The Dialog Corporation plc      All rights reserved.

      /H = Help            /L = Logoff          /NOMENU = Command Mode


Enter an option number to view information or to connect to an online
 service.  Enter a BEGIN command plus a file number to search a database
(e.g., B1 for ERIC).
?b corefull, coreabs

       03dec01 14:39:56 User242933 Session D72.1
            $0.00    0.233 DialUnits FileHomeBase
     $0.00  Estimated cost FileHomeBase
     $0.01  TYMNET
     $0.01  Estimated cost this search
     $0.01  Estimated total session cost   0.233 DialUnits


SYSTEM:OS  - DIALOG OneSearch
  File  15:ABI/Inform(R)  1971-2001/Dec 01
         (c) 2001 ProQuest Info&Learning
  File   9:Business & Industry(R)  Jul/1994-2001/Nov 30
         (c) 2001 Resp. DB Svcs.
  File 623:Business Week  1985-2001/Nov 30
```

```
        (c) 2001 The McGraw-Hill Companies Inc
  File 810:Business Wire  1986-1999/Feb 28
        (c) 1999 Business Wire
  File 275:Gale Group Computer DB(TM)  1983-2001/Nov 29
        (c) 2001 The Gale Group
  File 624:McGraw-Hill Publications  1985-2001/Dec 03
        (c) 2001 McGraw-Hill Co. Inc
  File 636:Gale Group Newsletter DB(TM)  1987-2001/Nov 30
        (c) 2001 The Gale Group
  File 621:Gale Group New Prod.Annou.(R)  1985-2001/Nov 30
        (c) 2001 The Gale Group
  File 813:PR Newswire  1987-1999/Apr 30
        (c) 1999 PR Newswire Association Inc
  File  16:Gale Group PROMT(R)  1990-2001/Nov 30
        (c) 2001 The Gale Group
  File 160:Gale Group PROMT(R)  1972-1989
        (c) 1999 The Gale Group
  File 148:Gale Group Trade & Industry DB  1976-2001/Nov 30
        (c)2001 The Gale Group
  File  20:World Reporter  1997-2001/Dec 03
        (c) 2001 The Dialog Corporation
  File  77:Conference Papers Index  1973-2001/Nov
        (c) 2001 Cambridge Sci Abs
  File  35:Dissertation Abs Online  1861-2001/Nov
        (c) 2001 ProQuest Info&Learning
  File 593:KOMPASS Central/Eastern Europe  2001/Sep
        (c) 2001 KOMPASS Intl.
  File  65:Inside Conferences  1993-2001/Dec W1
        (c) 2001 BLDSC all rts. reserv.
*File  65: For variance in UDs please see Help News65.
  File   2:INSPEC  1969-2001/Dec W1
        (c) 2001 Institution of Electrical Engineers
  File 233:Internet & Personal Comp. Abs.  1981-2001/Nov
        (c) 2001 Info. Today Inc.
  File  99:Wilson Appl. Sci & Tech Abs  1983-2001/Sep
        (c) 2001 The HW Wilson Co.
  File 473:FINANCIAL TIMES ABSTRACTS  1998-2001/APR 02
        (c) 2001 THE NEW YORK TIMES
*File 473: This file will not update after March 31, 2001.
It will remain on Dialog as a closed file.
  File 474:New York Times Abs  1969-2001/Nov 30
        (c) 2001 The New York Times
  File 475:Wall Street Journal Abs  1973-2001/Nov 30
        (c) 2001 The New York Times

      Set  Items  Description
      ---  -----  -----------
?s legend and (component or components) and (system or building or management or suppor
t) and ((texture or color or shading) 5n coding)
>>>Invalid syntax
?Cs legend and (component or components) and (system or building or management or suppo
rt) and (texture or color or shading)
>>>Invalid parameter:
?s legend and (component or components) and (system or building or management or suppor
t) and (texture or shading or color)
Processing
Processing
Processed  10 of  23 files ...
Processing
Processed  20 of  23 files ...
Completed processing all files
           112702  LEGEND
          1402579  COMPONENT
          2033432  COMPONENTS
         10117767  SYSTEM
          3699589  BUILDING
          9799414  MANAGEMENT
         6495261  SUPPORT
          124965  TEXTURE
           21608  SHADING
          909861  COLOR
      S1     751  LEGEND AND (COMPONENT OR COMPONENTS) AND (SYSTEM OR
                  BUILDING OR MANAGEMENT OR SUPPORT) AND (TEXTURE OR
                  SHADING OR COLOR)
?s s1 and (texture or shading or color) (w) coding
             751  S1
          124965  TEXTURE
           21608  SHADING
          909861  COLOR
          195164  CODING
            4954  ((TEXTURE OR SHADING) OR COLOR)(W)CODING
      S2       9  S1 AND (TEXTURE OR SHADING OR COLOR) (W) CODING
?type s2/3,ab/all
>>>No matching display code(s) found in file(s): 65, 593, 623-624, 810, 813
```

Considered all [illegible]

2/3,AB/1   (Item 1 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)
(c) 2001 ProQuest Info&Learning. All rts. reserv.

02269403 89297617
Showing what you know
Weber, Bruce R
Appraisal Journal v69n4 PP: 431-448 Oct 2001 ISSN: 0003-7087
JRNL CODE: APJ
WORD COUNT: 5160

ABSTRACT: A presentation on the use of geographic information systems is given. How an appraiser or appraisal reviewer could use GIS to find cases of real estate fraud is shown.

2/3,AB/2   (Item 2 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)
(c) 2001 ProQuest Info&Learning. All rts. reserv.

02052335 57511263
The role of GIS imaging in assessment administration
Skaff, Michael S; Murphy, Joseph G
Assessment Journal v7n3 PP: 23-29 May/Jun 2000 ISSN: 1073-8568
JRNL CODE: ASJ
WORD COUNT: 3131

ABSTRACT: Geographic information **system** (GIS) imaging is an effective assessment tool, both in facilitating the appraisal process and improving community relations in the appeal process. GIS technology uses geographic information to relate, collect, and edit a variety of information, which is then displayed on a computer screen in map form with superimposed numeric and symbolic data. Implementation of GIS imaging may reduce the number of field visits, as well as provide visual data to improve the accuracy of the appraisal process.

2/3,AB/3   (Item 3 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)
(c) 2001 ProQuest Info&Learning. All rts. reserv.

01729948 03-80938
The landscape of labor law enforcement in North America: An examination of Mexico's labor regulatory policy and practice
McGuinness, Michael Joseph
Law & Policy in International Business v29n3 PP: 365-413 Spring 1998
ISSN: 0023-9208 JRNL CODE: LPI

WORD COUNT: 19503

ABSTRACT: Critics of increased North American economic integration argued that Mexico's weak labor regulatory structure would lead to wide-scale social dumping in North America. The US, Canada, and Mexico proposed the North American Agreement on Labor Cooperation (NAALC), with its requirement that all NAFTA members enforce their domestic labor law, as a response to such arguments. This article describes Mexico's onsite labor inspection policy and practice, which represents the heart of labor law enforcement. It is through inspection that the governments of Mexico, the US, and Canada seek to monitor and promote compliance with the provisions of their labor law. A study of workplace inspection can provide an essential **component** of an informed and constructive debate concerning establishment of a continent-wide labor policy.

2/3,AB/4 (Item 4 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)
(c) 2001 ProQuest Info&Learning. All rts. reserv.

01551810 02-02799
Performance engineering of object-oriented systems
Smith, Connie U; Williams, Lloyd G
Capacity Management Review v25n11 PP: 1-18 Nov 1997 ISSN: 1049-2194
JRNL CODE: PPR
WORD COUNT: 5932

ABSTRACT: It is possible to design object-oriented systems that have adequate performance and exhibit the other qualities, such as reusability, maintainability, and modifiability, that have made object-oriented development (OOD) so successful. However, doing this requires careful attention to performance goals throughout the life cycle. The use of a performance modeling tool that supports the Software Performance Engineering process is described, for early life cycle evaluation of object-oriented systems. The evaluation of object-oriented software is illustrated with a simple example. Object-oriented methods will likely be the preferred design approach of the future. SPE techniques are vital to ensure that these systems meet performance requirements.

2/3,AB/5 (Item 5 from file: 15)
DIALOG(R)File 15:ABI/Inform(R)
(c) 2001 ProQuest Info&Learning. All rts. reserv.

00743936 93-93157
Coupling GIS with CAMA data in Johnson County, Kansas
Hensley, Tim
Property Tax Journal v12n1 PP: 19-35 Mar 1993 ISSN: 0731-0285
JRNL CODE: PTJ
WORD COUNT: 5714

ABSTRACT: Johnson County, Kansas, which includes the Kansas City metropolitan area, started developing a geographic information **system** (GIS) in 1985. Like so many jurisdictions, Johnson County experienced its fair share of problems, but by 1992, the appraisal office suddenly found at its fingertips a highly detailed, accurate, and comprehensive automated mapping **system** . Today the county planning office maintains its Automated Information/Mapping **System** (AIMS) with ARC/INFO software. Many map products and formats have been devised to assist the appraisal staff in their day-to-day activities of identifying, listing and valuing properties. A GIS linked with the county's computer-assisted mass appraisal (CAMA) **system** provides an effective tool for identifying properties and for analyzing location factors, improvement characteristics, and the ratio datum that commonly affects property values. In cartographic terms, this is thematic mapping.

2/3,AB/6 (Item 1 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)
(c) 2001 The Gale Group. All rts. reserv.

02168898 SUPPLIER NUMBER: 20424697 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Site Building.**(Caravelle's WebWatcher Java, Optimal Networks' Optical Application Insight, BMC Software's Patrol Knowledge Module for Internet Servers and Mercury Interactive's Astra SiteManager site** management **tools) (Software Review)(Evaluation)**
Linthicum, David S.
Computer Shopper, v18, n4, p530(1)
April, 1998
DOCUMENT TYPE: Evaluation ISSN: 0886-0556 LANGUAGE: English
RECORD TYPE: Fulltext
WORD COUNT: 1575 LINE COUNT: 00135

2/3,AB/7 (Item 2 from file: 275)
DIALOG(R)File 275:Gale Group Computer DB(TM)
(c) 2001 The Gale Group. All rts. reserv.

01348312 SUPPLIER NUMBER: 08146080 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Investigating the effects of color. **(effect of** color **on decision maker's ability to extract information from presentations) (technical)**
Hoadley, Ellen D.
Communications of the ACM, v33, n2, p120(7)
Feb, 1990
DOCUMENT TYPE: technical ISSN: 0001-0782 LANGUAGE: ENGLISH
RECORD TYPE: FULLTEXT; ABSTRACT
WORD COUNT: 4617 LINE COUNT: 00388

ABSTRACT: The results of a laboratory experiment designed to determine whether **color** improves the effectiveness of graphical and tabular data presentations is presented. Current MIS literature indicates that **color** improves performance in recall, retention, search-and-locate, and decision judgment tasks and increases comprehension of training materials. Pie charts and bar graphs are presented to decision makers in both multicolor (**color**) and monocolor (mono) treatments. Each subject views treatments and answers questions regarding the material, and the time it takes each to answer is measured. Statistical analysis of the time data shows that **color** improves time performance for bar graphs and pie charts. A small deterioration in time performance occurs with line graphs, but this difference is not statistically significant.

2/3,AB/8 (Item 1 from file: 624)
DIALOG(R)File 624:McGraw-Hill Publications
(c) 2001 McGraw-Hill Co. Inc. All rts. reserv.

0215763
Navajo Manuals Contain Serious Errors on Aileron Hook-ups, NTSB Finds
Regional Aviation Weekly May 4, 1990; Pg 166; Vol. 5, No. 18
Journal Code: RA ISSN: 1044-9450
Word Count: 595 *Full text available in Formats 5, 7 and 9*

2/3,AB/9 (Item 1 from file: 148)
DIALOG(R)File 148:Gale Group Trade & Industry DB
(c)2001 The Gale Group. All rts. reserv.

14053861 SUPPLIER NUMBER: 80195046 (USE FORMAT 7 OR 9 FOR FULL TEXT)
Showing What You Know.
Weber, Bruce R.
Appraisal Journal, 69, 4, 431
Oct, 2001
ISSN: 0003-7087 LANGUAGE: English RECORD TYPE: Fulltext

```
WORD COUNT:   5358    LINE COUNT:  00411
?type s2/9/4
```

2/9/4   (Item 4 from file: 15)

01551810 02-02799

Performance engineering of object-oriented systems

Smith, Connie U; Williams, Lloyd G

Capacity Management Review v25n11 PP: 1-18 Nov 1997 CODEN: EDPRDQ
ISSN: 1049-2194 JRNL CODE: PPR
DOC TYPE: Journal article LANGUAGE: English   LENGTH: 17 Pages
SPECIAL FEATURE: Charts References
WORD COUNT: 5932

ABSTRACT: It is possible to design object-oriented systems that have adequate performance and exhibit the other qualities, such as reusability, maintainability, and modifiability, that have made object-oriented development (OOD) so successful. However, doing this requires careful attention to performance goals throughout the life cycle. The use of a performance modeling tool that supports the Software Performance Engineering process is described, for early life cycle evaluation of object-oriented systems. The evaluation of object-oriented software is illustrated with a simple example. Object-oriented methods will likely be the preferred design approach of the future. SPE techniques are vital to ensure that these systems meet performance requirements.

TEXT: Our experience has shown that it is possible to design object-oriented systems that have adequate performance and exhibit the other qualities, such as reusability, maintainability, and modifiability, that have made Object-oriented development (OOD) so successful. However, doing this requires careful attention to performance goals throughout the life cycle. This article describes the use of a performance modeling tool that supports the Software Performance Engineering (SPE) process, for early life cycle evaluation of object-oriented systems. The evaluation of object-oriented software is illustrated with a simple example.

Introduction.

Object-oriented development methods have been shown to be valuable in constructing software systems that are easy to understand and modify, have a high potential for reuse, and are relatively quick and easy to implement. Despite the demonstrated successes of OOD, many organizations have been reluctant to adopt object-oriented techniques, largely due to concerns over performance.

Our experience has shown that it is possible to design object-oriented systems that have adequate performance and exhibit the other qualities, such as reusability, maintainability, and modifiability, that have made OOD so successful [Smith and Williams, 1993]. However, doing this requires careful attention to performance goals throughout the life cycle. Failure to build-in performance from the beginning can result in the need to "tune" code, destroying the benefits obtained from a careful objectoriented design. In addition, it is unlikely that "tuned" code will ever equal the performance of code that has been engineered for performance. In the worst case, it will be impossible to meet performance goals by tuning, necessitating a complete redesign or even cancellation of the project.

SPE for object-oriented systems is especially difficult since functionality is decentralized. Performing a given function is likely to require collaboration among many different objects from several classes. These interactions can be numerous and complex and are often obscured by polymorphism and inheritance, making them difficult to trace. Distributing objects over a network can compound the problem.

One of the principal barriers to the effective use of SPE with OOD is the gap between the designers who need feedback on the performance implications of design decisions and the performance specialists who have the skill to conduct comprehensive performance engineering studies with typical modeling tools. This gap means that extra time and effort is required to coordinate design formulation and analysis, effectively limiting the ability of designers to explore design alternatives.

The ideal long-term solution to providing SPE assessments during the design stage is an evolution of today's Computer-aided Software Engineering (CASE) tools to provide decision **support** for many facets of the design including correctness, completeness, performance, reliability, and so on. This approach, however, is not currently practical. It is too expensive for each CASE vendor to create their own modeling/analysis **component** . Therefore, we seek a near-term capability to interface CASE tools to existing modeling tools. A previous article defined the SPE information that CASE tools must collect [Williams and Smith, 1995]. This article illustrates the translation from object-oriented design models into performance models, and the early life cycle performance evaluation of object-oriented systems.

The article begins by reviewing related work and is followed by an overview of features of software performance modeling tools necessary to facilitate the evaluation of object-oriented and other software systems. We then present an overview of the process of software performance engineering for object-oriented systems. A simple example illustrates the process.

Related work.

Object-oriented methods typically defer consideration of performance issues until detailed design or implementation (see e.g., [Rumbaugh, et al., 1991], [Booch, 1994]). Even then, the approach tends to be very general. There is no attempt to integrate performance engineering into the development process.

Some work specifically targeted at object-oriented systems has emerged from the performance community. Smith and Williams [Smith and Williams, 1993] describe performance engineering of an object-oriented design for a real time **system** . However, this approach applies general SPE techniques and only addresses the specific problems of object-oriented systems in an ad hoc way.

Hrischuk et. al., [Hrischuk, et al., 1995] describe an approach based on constructing an early prototype which is then executed to produce angio traces. These angio traces are then used to construct workthreads (also known as timethreads or use case maps [Buhr and Casselman, 1992], [Buhr and Casselman, 1994], [Buhr and Casselman, 1996]), which are analogous to execution graphs. Workthreads provide empirical information about traversal frequencies for data-dependent choices and loops. Service times are estimated. This differs from the approach described here in that scenarios are derived from prototype execution rather than from the design and the **system** execution model is then generated automatically from the angio traces.

Baldassari et.al., propose an integrated object-oriented CASE tool for software design that includes a simulation capability for performance assessment [Baldassari, et al., 1989], [Baldassari and Bruno, 1988]. The CASE tool uses petri nets for the design description language rather than the general methods described above, thus the design specification and the performance model are equivalent and no translation is necessary. Using these capabilities requires developers to use both the PROTOB method and CASE tool.

This article uses the SPE tool SPEED to conduct the performance analysis. Other software modeling tools are available, such as [Beilner, et al., 1988], [Beilner, et al., 1995], [Goettge, 1990], [Grummit, 1991], [Rolia, 1992]. The approach described here could be adapted to other tools. Adaptation is necessary for these other tools that do not use execution graphs as their model paradigm.

Software performance modeling overview.

This section gives a brief overview of the desirable features of an SPE tool that make it appropriate for OOD (and other) evaluations throughout their development life cycle.

Focus. With a software performance modeling tool, users create graphical models of envisioned software processing and provide performance specifications. **System** execution models (queuing networks) are automatically generated from the software model specifications. A combination of analytic and simulation model solutions identify potential performance problems and software processing steps that may cause the problems.

The goal is to use the simplest possible model that identifies problems with the software architecture, design, or implementation plans. Simple models are desired because in the early life cycle phase in which they are created:

developers seldom have exact data that justifies a more sophisticated model,

they need quick feedback to influence development decisions,

they need to comprehend the model results, especially the correlation of the software decisions to the computer resource impacts.

Model description. The user's view of the model is a scenario, an execution graph of the software processing steps [Smith, 1990]. Software scenarios are assigned to the facilities that execute the processing steps. Models of distributed processing systems may have many scenarios and many facilities. Users specify software resource requirements for each processing step. Software resources may be the number of messages transmitted, the number of SQL queries, the number of SQL updates, etc. depending on the type of **system** to be studied and the key performance drivers for that **system** . A performance specialist provides overhead specifications that specify an estimate of the computer resource requirements for each software resource request. These are specified once and reused for analysis of all software that executes in that environment. This step is described in more detail later.

Model solution. The simple models benefit from a combination of model solutions: analytic results for the software models, and an approximate, analytic solution of the generated **system** execution model. A simulation solution is appropriate later in the development cycle for complex models with multiple software scenarios executing on one or more computer **system** facilities. The user should select the type of solution appropriate for the development life cycle stage and thus the precision of the data that feeds the model. There is no need for a detailed, lengthy simulation when only rough guesses of resource requirements are specified.

Model results. The results needed to evaluate software performance models include: the end-to-end response time, the elapsed time for each processing step, the device utilization, and the amount of time spent at each computer device for each processing step. This identifies both the potential computer device bottlenecks, and the portions of the device usage by processing step (thus the potential software processing bottlenecks).

In addition to numeric values for results, some visual interpretation of results helps developers to comprehend the results and correlate **system** resource usage to software processing steps. For example, model results presented both with numeric values and **color coding** that uses cool colors (blue and green) to represent relatively low values and hot colors (yellow and red) calls attention to relatively high values.

Application areas. Software performance models have the greatest impact when modeling software systems under development. Some software systems include: operating systems, database **management** systems, or custom applications. The software may be evaluated on any hardware/software platform combination. The software may execute on a uniprocessor or in a distributed or Client/Server environment. The environment and platforms are represented in the **system** execution model. Software performance models are also useful for modeling existing systems to study scalability, enhancements, etc.

SPE process steps for OOD.

The process for performing SPE for an object-oriented design begins with a set of scenarios. A scenario is a description of the interactions between the **system** and its environment or between the internal objects involved in a particular use of the **system** under development. The scenario shows the objects that participate and the messages that flow between them. A message may represent either an event or invocation of one of the receiving object's operations.

The use of scenarios has become popular in many current approaches to object-oriented development. Scenarios, known as use cases, are an important **component** of Jacobson's Objectory Method [Jacobson, et al., 1992]. Scenarios are also used in OMT [Rumbaugh, et al., 1991], Booch [Booch, 1994], Fusion [Coleman, et al., 1994], and the new Unified Modeling Language [Booch and Rumbaugh, 1995]. In object-oriented methods, scenarios are used to:

describe the externally visible behavior of the **system** ,

involve users in the requirements analysis process, **support** prototyping,

help validate the requirements specification,

understand interactions between objects, and

support requirements-based testing.

Once the major functional scenarios have been identified, those that are important from a performance perspective are selected for performance modeling. Scenarios that are important to performance can be identified by a variety of techniques, including experience with similar systems and performance walkthroughs [Smith, 1990].

The scenarios are then translated to execution graph software performance models (see below) that provide the performance results. Currently, this translation is manual. However, the close correspondence between the way scenarios are expressed in object-oriented methods and execution graphs suggests that an automated translation should be possible.

The next SPE steps are conducted after the translated model is entered into a software performance modeling tool. Performance engineers enter data for the processing steps in the execution graphs, ensure that correct overhead specifications are provided, and evaluate model solutions for alternatives. These steps are illustrated in the next section.

An example.

To illustrate the use of software performance models for evaluating the performance of object-oriented systems, we present an example based on a simple automated teller machine (ATM).

The ATM accepts a bank card and requests a personal identification number (PIN) for user authentication. Customers can perform any of three transactions at the ATM including (1) deposit cash to an account, (2) withdraw cash from an account, or (3) request the available balance in an account. A customer may perform several transactions during a single ATM session. The ATM communicates with a computer at the host bank which verifies the account and processes the transaction. When the customer is finished using the ATM, a receipt is printed for all transactions and the customer's card is returned.

Here, we focus on scenarios that describe the use of the ATM. A full specification would include additional models, such as a class diagram and behavior descriptions for each class. However, our interest here is primarily in the use of scenarios as a bridge between Object-Oriented Development and Software Performance Engineering. Thus, these additional models are omitted.

Scenarios. As described in [Williams and Smith, 1995], scenarios represent a common point of departure between objectoriented requirements or design models and SPE models. Scenarios may be represented in a variety of ways [Williams, 1994]. Here, we use Message Sequence Charts (MSCs) to describe scenarios in object-oriented models. The MSC notation is specified in ITU standard Z.120 [ITU, 1996]. Several other notations used to represent scenarios are based on MSCs (examples include: Event Flow Diagrams [Rumbaugh, et al., 1991]; Interaction Diagrams Jacobson, et al., 1992], [Booch, 1994]; and Message Trace Diagrams [Booch and Rumbaugh, 1995]). However, none of these incorporates all of the features of MSCs needed to establish the correspondence between object-oriented scenarios and SPE scenarios.
(Chart Omitted)

Captioned as: FIGURE 1.

(Chart Omitted)

Captioned as: FIGURE 2.

Figure 1 on page 7 illustrates a high-level MSC for the ATM example. Each object that participates in the scenario is represented by a vertical line or axis. The axis is labeled with the object name (e.g., anATM). The vertical axis represents relative time which increases from top to bottom; an axis does not include an absolute time scale. Interactions between objects (events or operation invocations) are represented by horizontal arrows.

Figure 1 describes a general scenario for user interaction with the ATM. The rectangular areas labeled "loop" and "alt" are known as inline expressions and denote repetition and alternation. This Message Sequence Chart indicates that the user may repeatedly select a transaction which may be a deposit, a withdrawal, or a balance inquiry. The rounded rectangles are "MSC references" that refer to other MSCs. The use of MSC references allows horizontal expansion of Message Sequence Charts. The MSC that corresponds to Process Withdrawal is shown in Figure 2.

A Message Sequence Chart may also be decomposed vertically, i.e., a refining MSC may be attached to an instance axis. Figure 3 shows a part of the decomposition of the anATM instance axis. The dashed arrows represent object instance creation or destruction.

Mapping scenarios to performance models. Models for evaluating the performance characteristics of the proposed ATM **system** are based on performance scenarios for the major uses of the **system** . These performance scenarios are the same as the functional scenarios illustrated in the message sequence charts as shown in Figures 1 through 3. However, they are represented using Execution Graphs. Note that not all functional scenarios are necessarily significant from a performance perspective. Thus, an SPE study would only model those scenarios that represent user tasks or events that are significant to the performance of the **system** .

Figure 4 on page 10 shows an Execution Graph illustrating the general ATM scenario. The case node indicates a choice of transactions while the repetition node indicates that a session may consist of multiple transactions. Subgraphs corresponding to the expanded nodes show additional processing details. The processing steps (basic nodes) correspond to steps in the lowest-level Message Sequence Chart diagram for the scenario. The execution graph in Figure 4 shows an end-to-end session that spans several ATM customer interactions. Thus analysts can evaluate the performance for each individual customer interaction as well as the total time to complete a session. (Note: Some performance analysts prefer to evaluate a traditional transaction - the processing that occurs after a user presses the Enter key until a response appears on the screen. This eliminates the highly variable, userdependent time it takes to respond to each prompt. While that approach was appropriate for mainframe transaction based applications, the approach prescribed here is better for Client/Server and other distributed systems with graphical user interfaces. The screen design and user interaction patterns may introduce end-to-end response time problems even though computer resource utilization is low.)

Performance evaluation. After identifying the scenarios and their processing steps in the MSC, the analyst uses a software performance modeling tool to create and evaluate the execution graph model. This article illustrates the use of SPEED, a performance modeling tool that supports the SPE process described in [Smith, 1990]. Figure 5 on page 11 shows the tool's screen. The "world view" of the model appears in the small navigation boxes on the right side of the screen. The correspondence between an expanded node and its subgraph is shown through **color** . For example, the top level of the model is in the top-left navigation box; its nodes are black. The top-right navigation box (turquoise) contains the loop to get the transaction and process it. Its corresponding expanded node in the top-level model is also turquoise. The Process Withdrawal subgraph is in the large area of the screen (and in the second row, left navigation box). Users can directly access any level in the model by clicking on the corresponding navigation box.

(Chart Omitted)

Captioned as: FIGURE 3.

The next step is to specify software resource requirements for each processing step. The software resources we examine for this example are:

Screens. The number of screens displayed to the ATM customer (aUser),

Home. The number of interactions with the hostBank,

Log. The number of log entries on anATM machine, and
Delay. The relative delay for the ATM customer (aUser) to respond to a prompt, or the time for other ATM device processing such as the cash dispenser or receipt printer.

Up to five types of software resources may be specified. The set of five may differ for each subgraph if necessary to characterize performance. The user provides values for these requirements for each processing step in the model, as well as the probability of each case alternative and the number of loop repetitions. The specifications may include parameters that can be varied between solutions, and may contain arithmetic expressions. Resource requirements for expanded nodes are in the processing steps in the corresponding subgraph.

The software resource requirements for the Process Withdrawal subgraph are illustrated in Figure 5. Note that the DispenseCash step displays a screen to inform the customer to take the cash, logs the action to the ATM's disk, and has a delay for the cash dispenser. We arbitrarily assume this delay to be 5 time units. In the software model the delay is relative to the other processing steps; e.g., the delay for the customer to remove the cash is twice as long as DispenseCash. The user may specify the duration of a time unit (in the overhead matrix) to evaluate the effect on overall performance; e.g., a range of .1 sec. to 2 sec. per time unit. In this example a time unit is 1 sec.

The ATM scenario focuses on processing that occurs on the ATM. However, the performance of anATM unit is seldom a performance problem. The evaluation will examine the performance at a hostBank that supports many ATM units.

(Chart Omitted)

Captioned as: FIGURE 4.

Processing overhead. Analysts specify values for the software resource requirements for processing steps. The computer resource requirements for each software resource request are specified in an overhead matrix stored in the SPE database. This matrix is used for all software models that execute in that hardware/software environment. Figure 6 on page 14 shows the overhead matrix for this case study. The matrix connects the values specified in the "ATM Spec Template" software specification template with the device usage in the "Host Bank" computer facility. The software resources in the template are in the left column of the matrix; the devices in the facility are in the other columns. The values in the matrix describe the device characteristics.
The pictures of the devices in the facility are across the top of the matrix, and the device name is in the first row. The second row specifies how many devices of each type are in the facility. For example, if the facility has 20 disk devices, there is one disk device column with 20 in its quantity row. The (deliberately) simple models will assume that disk accesses can be evenly spread across these devices. The third row is a comment that describes the service units for the values specified for the software processing steps. The next five rows are the software resources in the specification template. This example uses only four of them. The last row specifies the service time for the devices in the computer facility.

(Chart Omitted)

Captioned as: FIGURE 5.

The values in the center section of the matrix define the connection between software resource requests and computer device usage. The screen display occurs on anATM unit; its only affect on the hostBank is a delay. The 1 in the ATM column for the "Screen" row means that each screen specified in the software model causes one visit to the ATM delay server. We arbitrarily assume this delay to be one second (in the service time row). Similarly, each log and delay specification in the software model result in a delay between hostBank processing requests. We assume the log delay is 0.01 seconds. The delays due to processing at the ATM unit could be calculated by defining the overhead matrix for the ATM facility and solving the scenario to calculate the time required.

This example assumes that all ATM transactions are for this hostBank; they do not require remote connections. This version of the model assumes 1500 K Instructions execute on the host bank's CPU (primarily for data base accesses), 8 physical I/Os are required, and a delay of 0.1 seconds for the network transmission to the host bank. These values may be measured, or estimates could be obtained by constructing and evaluating more detailed models of the host processing required.

Thus each value specified for a processing step in the software model generates demand for service from one or more devices in a facility. The overhead matrix defines the devices used and the amount of service needed from each device. The demand is the product of the software model value times the value in the overhead matrix cell times the service time for the column.

Model solutions and results. The analyst first solves a "No Contention" model to confirm that in the best case, a single ATM session will complete in the desired time, without causing performance bottlenecks at the host bank. Up to four sets of results may be displayed concurrently, as shown in Figure 7 below.

The elapsed time result for the "No Contention" model is in the topleft quadrant. The overall time is at the top, and the time for each processing step is next to the step. The **color** bar **legend** in the upper right corner of the quadrant shows the values associated with each **color** ; the upper bound is set by defining an overall performance objective. Values higher than the performance objective will be red, lower values are respectively cooler colors. The "Resource usage" values below the **color** bar **legend** show the time spent at each computer device. Of the 35.6 total seconds for the end-to-end scenario, 35.25 is due to the delays at the ATM unit for customer interactions and processing. Thus, no performance problems are apparent with this result.

(Chart Omitted)

Captioned as: FIGURE 6.

(Chart Omitted)
Captioned as: FIGURE 7.

(Chart Omitted)

Captioned as: FIGURE 8.

The SPE tool evaluates the results of device contention delays by automatically creating and solving an analytic queuing network model. The utilization result for the "Contention solution" of the ATM sessions with an arrival rate of 5 withdrawal transactions per second is in the topright quadrant of Figure 7. The total utilization of each server is shown under the **color** bar, and the utilization of each device by each processing step is next to the step. The total CPU utilization is 15%, and the disk device is 100%. Even though the customer data base would fit on one disk device, more are needed to relieve the contention delays. In general, options for correcting bottlenecks are to reduce the number of I/Os to the disk, reduce the number of ATM units that share a host bank server, or add disks to the server. The options are evaluated by changing software processing steps, or values in the overhead matrix. The results in the lower quadrants of Figure 7 show the utilization and response time results for 3 disk devices. The quadrants let the analyst easily compare performance metrics for alternatives.
System execution model. The "**System** model solution" creates a queuing network model with all the scenarios defined in a project executing on all their respective facilities. The **system** execution model picture is in Figure 8 at left. This example models one workload scenario: a session with one withdrawal transaction. The host bank may have other workloads such as teller transactions, bank analyst inquiries, etc. Each performance scenario in the SPE database appears across the top of the **system** execution model screen. The specification template beside the scenario name displays the current workload intensity and priority. Below the scenario is a template that represents the devices in the facility assigned to the scenario. The facilities in the SPE database appear across the bottom of the **system** execution model screen. This example models only one facility for the host bank. It could also model the ATM unit, other home banks, etc.

The model is solved by calculating the model parameters from the software model for each scenario and then automatically constructing and solving a CSIM simulation model [Schwetman, 1994]. CSIM is a simulation product that is widely used to evaluate distributed and parallel processing systems. The model results show:

the response time for the scenario and its corresponding **color** (inside the scenario name rectangle),
the amount of the total response time spent at each computer device (the values and colors in the template below the scenario name),

the average utilization of each device in each facility and its corresponding **color** .

One of the model scenarios may be a focus scenario, usually a scenario that is vital to the **system** development project. Users may view various

system model results for processing steps in the focus scenario in 2 quadrants below the **system** model.

One of the difficult problems in simulation is determining the length of the simulation run. SPEED solves this problem with the performance results meter shown in Figure 9 below left. It is based on work by [Raatikainen, 1993] adapted for SPE evaluations. The approach is adapted because results depend on the precision of the model parameters, and in early life cycle stages only rough estimates are available. Users may monitor simulation progress and stop the simulation to view results at any point. They may also set a confidence value or simulated run time to automatically terminate the simulation. The meter shows the progress of the current simulation; it is calculated with a batch-mean method. The tool uses a default value of 70% probability that the reported response is within 30% of the actual mean. This bound is much looser than typically used for computer **system** models. We selected this value empirically by studying the length of time required for various models, versus the analytic performance metrics, versus the precision of the specifications that determine the model parameters. Users may increase the confidence value, but not the probability. It would be easy to add other controls, but experience with our users shows that this approach is sufficient for their evaluations and requires little expert knowledge of simulation controls.

Summary.

This article describes the use of software performance models for performance engineering of object-oriented software. It describes how to use scenarios to determine the processing steps to be modeled, and illustrates the process with a simple ATM example defined with Message Sequence Charts. It then illustrates the SPE evaluation steps of the derived software performance model.

Object-oriented methods will likely be the preferred design approach of the future. SPE techniques are vital to ensure that these systems meet performance requirements. SPE for OOD is especially difficult since functions may require collaboration among many different objects from many classes. These interactions may be obscured by polymorphism and inheritance, making them difficult to trace. Distributing objects over a network compounds the problem. Our approach of connecting performance models and designs with message sequence charts makes SPE performance modeling of object-oriented software practical. A performance modeling tool makes it easier for software designers to conduct their own performance studies. Features that de-skill the performance modeling process and make this viable are:

quick and easy creation of performance scenarios,

automatic generation of **system** execution models,

visual perception of results that call attention to potential performance problems,

(Chart Omitted)

Captioned as: FIGURE 9.

simulation length control that can be adapted to the precision of the model input data.
Other features **support** SPE activities other than modeling such as SPE database archives, and presentation and reporting of results. Once performance engineers complete the initial SPE analysis with the simple models and ensure that the design approach is viable, they may export the models for use by "industrial strength" performance modeling tools [Smith and Williams, 1995].

As noted in earlier, Message Sequence Charts do not explicitly capture time. However, the close structural correspondence between scenarios expressed in Message Sequence Charts and those using Execution Graphs suggests the possibility of a straightforward translation from analysis/design models to performance scenarios. Standard SPE techniques, such as performance walkthroughs, best-and-worst-case analysis, and others, can then be used to obtain resource requirements or time estimates for processing steps.

Providing specifications for the overhead matrix still requires expert knowledge of the hardware/software processing environments and performance measurement techniques. Some performance modeling tools provide libraries with default values for the path lengths. If the benchmark studies that led to the library values closely match the new software being modeled, the default values are adequate. Someone must validate the default values with measurements to ensure that they apply to the new software. Thus the expert knowledge is necessary for both tool approaches. It would be relatively easy to build a feature to automatically populate an overhead matrix from customized measurement tool output.

This article demonstrates the feasibility of applying SPE to objectoriented systems. Future research is aimed at providing a smooth transition between CASE tools for OOD and SPE evaluation tools. X

Reference:

References.

Reference:

[Baldassari, et al., 1989] M. Baldassari, B. Bruno, V. Russi, and R. Zompi, "PROTOB: A Hierarchical Object-Oriented CASE Tool for Distributed Systems," European Software Engineering Conference Proceedings, 1989, Coventry, England, September, 1989. [Baldassari and Bruno, 1988] M. Baldassari and G. Bruno, "An Environment for Object-Oriented Conceptual Programming Based on PROT Nets," intdvances in Petri Nets, Lectures in Computer Science No. 340, Berlin, Springer-Verlag, 1988, pp. 1-19. [Beilner, et al., 1988] H. Beilner, J. Mater, and N. Weissenburg, "Towards a Performance Modeling Environment: News on HIT," 4th International Conference Proceedings on Modeling Techniques and Tools for Computer Performance Evaluation, Plenum Publishing, 1988.

Reference:

[Beilner, et al.,1995] H. Beilner, J. Mater, and C. Wysocki, "The Hierarchical Evaluation Tool HIT," in Performance Tools and Model Interchange Formats, Vol. 581/1995, F. Bause and H. Beilner, ed., Dortmund, Germany, Universitat Dortmund, Fachbereich Informatik, 1995, pp. 6-9.

[Booch,1994] G. Booch, Object-Oriented,Analysis and Design with Applications, Redwood City, CA, Benjamin/Cummings,1994. [Booch and Rumbaugh,1995] G. Booch and J. Rumbaugh, "Unified Method for Object-Oriented Development," Rational Software Corporation, Santa Clara, CA, 1995.

[Buhr and Casselman, 1996] R. J. A. Buhr and R. S. Casselman, Use Case Maps for Object-Oriented Systems, Upper Saddle River, NJ, Prentice Hall, 1996.

[Buhr and Casselman,1994] R.J. A. Buhr and R. S. Casselman, "Timethread-Role Maps for Object-Oriented Design of Real-Time and Distributed Systems," OOPSLA '94 Proceedings, Object-Oriented Programming Systems, Languages and Applications, Portland, OR, October, 1994, pp. 301-316.

[Buhr and Casselman, 1992] R. J. A. Buhr and R. S. Casselman, "Architectures with Pictures," OOPSLA '92 Proceedings, Objectoriented Programming Systems, Languages and Applications, Vancouver, BC, October,1992, pp. 466-483. [Coleman, et al.,1994] D. Coleman, P. Arnold, S. Bodoff, C. Dollin, H. Gilchrist, F. Hayes, and P. Jeremaes, Object-Oriented

Development: The Fusion Method, Englewood Cliffs, NJ, Prentice Hall,1994.

[Goettge,1990] R. T. Goettge, "An Expert **System** for Performance Engineering of Time-Critical Software," Computer Measurement Group Conference Proceedings, Orlando, FL, 1990, pp. 313-320.

Reference:

[Grummit,1991] A. Grummit, A Performance Engineer's View of Systems Development and Trials," Computer Measurement Group Conference Proceedings, Nashville, TN, 1991, pp. 455-463.
[Hrischuk, et al.,1995] C. Hrischuk,J. Rolia, and C. M. Woodside, "Automatic Generation ofa Software Performance Model Using an Object-Oriented Prototype," Third International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems Proceedings, Durham, NC,January,1995, pp. 399-409. [ITU,1996] ITU, "Criteria for the Use and Applicability of Formal Description Techniques, Message Sequence Chart (MSC)," International Telecommunication Union,1996.

[Jacobson, et al.,1992] I.Jacobson, M. Christerson, P.Jonsson, and G. Overgaard, Object-Oriented Software Engineering, Reading, MA, Addison-Wesley, 1992.

[Raatikainen, "Accuracy of Estimates for Dynamic Properties of Queuing Systems in Interactive Simulation," University of Helsinki, Department of Computer Science Teollisuuskatu 23, Helsinki, Finland,1993. [Rolla,1992] J. A. Rolia, "Predicting the Performance of Software Systems," Ph.D. Thesis, University of Toronto,1992. [Rumbaugh, et al.,1991] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen, Object-Oriented Modeling and Design, Englewood Cliffs, NJ, Prentice Hall, 1991.

[Schwetman,1994] H. Schwetman, "CSIM17: A Simulation Model- **Building** Tool,"Winter Simulation Conference Proceedings, Orlando, FL, 1994.
Reference:

[Smith, 1990] C. U. Smith, Performance Engineering of Software Systems, Reading, MA, Addison-Wesley,1990. [Smith and Williams,1995] C. U. Smith and L. G. Williams, "A Performance Model Interchange Format," in Performance Tools and Mode/Interchange Formats, Vol. 581/1995, F. Bause and H. Beilner, ed., Dortmund Germany, Universitat Dortmund, Informatik IV, 1995, pp. 67-85.

[Smith and Williams,1993] C. U. Smith and L. G. Williams, "Software Performance Engineering: A Case Study Including Performance Comparison with Design Alternatives," IEEE Transactions on Software Engineering, Vol.19, no. 7, pp. 720-741,1993. [Williams,,1994] L. G. Williams, G. Williams,"Definition of Information Requirements for Software Performance Engineering," Technical Report No. SERM-021-94, Software Engineering Research, Boulder, CO, October,1994.

[Williams and Smith,1995] L. G. Williams and C. U. Smith, "Information Requirements for Software Performance Engineering," in Quantitative Evaluation of Computing and Communication Systems, Lecture Notes in Computer Science, Vol. 977, H. Beilner and F. Bause, ed., Heidelberg, Germany, Springer-Verlag,1995, pp. 86-101.
Author Affiliation:

About the authors. Dr. Connie U. Smith, a principal consultant of Performance Engineering Services, is internationally known for her pioneering work in SPE With over 20 years experience, she specializes in seminars, consulting, and products to **support** SPE. Her research interests include software performance engineering, performance modeling, performance engineering for safety-critical systems, computer graphics, and objectoriented development. She first proposed the term software performance engineering in an award-winning 1981 paper. She is a past ACM National Lecturer, served as an officer of ACM SIGMETRICS, and is an active member of the Computer Measurement Group, having served on the CMG Board of

Directors for 4 years. Connie is a 1986 AA Michelson Award presented for technical excellence and professional contributions and listed in Who's Who of American Women and numerous other professional listings. Connie received a BA in mathematics from the University of Colorado and MA and Ph.D. degrees in computer science from the University of Texas at Austin.

Author Affiliation:

Dr. Lloyd G. Williams is the founder and President of Software Engineering Research. He is an internationally recognized authority on software methods, tools, and environments. His primary areas of expertise include software development methods, object-oriented development, software performance engineering, software tools (CASE), and software development environments. Lloyd presents professional development seminars and has consulted on software development for more than 100 organizations in the USA, Japan, and Europe. He received the A.B. degree from Colgate University and a Ph.D. in physical chemistry from the University of Wisconsin. Lloyd is a member of the ACM and the IEEE Computer Society. He is author of numerous technical articles, is an Associate Editor of the Journal of Systems and Soft,care, and is a contributor to the AIAA Progress Series book,Aerospace Software Engineering.

GEOGRAPHIC NAMES: US

DESCRIPTORS: Performance evaluation; Object oriented programming; Software; Studies

CLASSIFICATION CODES: 5240 (CN=Software & systems); 9130 (CN=Experimental/Theoretical)

?

	Type	L #	Hits	Search Text	DBs	Time Stamp
1	BRS	L1	0	legend and component and (system or building or management or support) and ((texture or color or shading) near coding)	EPO; JPO; DERWENT	2001/12/03 13:43
2	BRS	L2	4	legend and component and (system or building or management or support) and (texture or color or shading)	EPO; JPO; DERWENT	2001/12/03 13:44
3	BRS	FAMILY	1	SU-575666-$.DID.	DERWENT	2001/12/03 13:44

Considered [illegible]
Considered [illegible]
[illegible]

	Type	L #	Hits	Search Text	DBs	Time Stamp
1	BRS	L1	8066	legend and component and (system or building or management or support)	USPAT	2001/12/03 09:40
2	BRS	L2	2366	legend and component and (system or building or management or support) and (texture or color or shading)	USPAT	2001/12/03 09:42
3	BRS	L3	100	legend and component and (system or building or management or support) and ((texture or color or shading) near coding)	USPAT	2001/12/03 09:43

considd title [illegible]